UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/901,363	07/09/2001	Robert Hundt	10012768-1	5331

7590 07/01/2005

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, CO 80527-2400

EXAMINER	
RAMPURIA, SATISH	

ART UNIT	PAPER NUMBER
2191	

DATE MAILED: 07/01/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/901,363	HUNDT ET AL.	
	Examiner	Art Unit	
	Satish S. Rampuria	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *12 April 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-15* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-15* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

Response to Amendment

1. This action is in response to the appeal brief filed on 04/12/2005.

2. Claims 1-15 are pending.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

4. Claims 1, 2 and 15 are rejected under 35 U.S.C. 102(e) as being anticipated by US Patent No. 6,665,671 to Coutant et al., hereinafter called Coutant.

Per claim 1:

Coutant discloses:

- ***A computer-implemented method for optimizing an executable program*** (col. 4, lines 64-65 "data load optimizer 140... source program, executable program") ***having a plurality of functions*** (col. 4, line 66 "comprising a set of instructions") ***and at least one function with a first name associated with executable code that implements the function at a first address*** (col. 4, lines 43-46 "memory 42... include one or more... programs... comprises... listing of executable instructions for implementing logical functions") ***and at least one linkage stub code segment having code that branches to the first address*** (col. 7, lines 16-

18 "function B 83 within module A 81 references (address) data area DD 95 within module X 91, the function B 83 first references the linkage table 86 that was created by the linker program 54") ***and a symbolic name by which the function is invoked in the program*** (col. 8, lines 66-67 "dynamic loader 120 resolves the symbolic references in the symbol table")

- ***identifying branch instructions*** (col. 3, lines18-20 "in a function call, a jump/branch instruction is used to transfer control from one point in the code to another") having ***target addresses that reference the linkage stub code segment*** (col. 3, lines 24-26 "at load time with the actual address once the second load module has been loaded")
- ***replacing the target addresses of the branch instructions with the first address*** (col. 8, lines 26-27 "the data load optimizer 140 then replaces the linkage table load with the no-op instruction at step 145" see fig. 7)

Per claim 2:

The rejection of claim 1 is incorporated, and further, Coutant discloses:

- ***replacing the target address of the branch instructions with the first address only in functions*** (col. 8, lines 26-26 "the data load optimizer 140 then replaces the linkage table load with the no-op instruction at step 145" see fig. 7) ***that are reached during program execution*** (col. 8, lines 23-24 "if it is determined at step 144 that the global pointer relative offset is small enough to be handled")

Claim 15 is the apparatus claim corresponding to method claim 1 and rejected under the same rational set forth in connection with the rejection of claim 1 above.

5. Claims 1, 2 and 15 are rejected under 35 U.S.C. 102(e) as being anticipated by US Patent No. 6,006,033 to Heisch, hereinafter called Heisch.

Per claim 1:

Heisch discloses:

- ***A computer-implemented method for optimizing an executable program*** (col. 2, lines 38-39 "reordering the instructions in an executable program") ***having a plurality of functions*** (col. 5, line 13-14 "FIG. 3 is a table showing a plurality of actual instructions corresponding to a particular function written in source code") ***and at least one function with a first name associated with executable code that implements the function at a first address*** (col. 5, lines 3-5 "Trace tool 52 determines trace information, such as what functions the instructions perform and the address(es) at which the instructions are stored") ***and at least one linkage stub code segment having code that branches to the first address*** (col. 7, lines 58-60 "the branch reg 72 will cause the program to return to instruction 2 by a first branch along path 75... second branch then occurs from block 71 to block 76 due to the unconditional branch instruction B L2") ***and a symbolic name by which the function is invoked in the program*** (col. 7, lines 62-67 "the vast majority of DCBs found in most executable programs are due to 1) the C switch/case statement (which typically generates a branch table in the program constant area)")
- ***identifying branch instructions*** (col. 3, lines18-20 "a search of the program for the next branch instruction at step 81") having ***target addresses that reference the linkage stub code segment*** (col. 6, lines 20-23 "conditional branch instruction code has been rewritten with a

different branch target address and the opposite (reversed sense) condition code (from a BNE Target.sub.-- Address to BEQ Fall.sub.-- Thru.sub.-- Address).")

- ***replacing the target addresses of the branch instructions with the first address*** (col. 7, lines 27-31 "Reordered instructions are appended to the end of the original executable (in the "reordered text area") and are replaced (in the "original text area") with branches to the addresses of where the instructions have been moved).

Per claim 2:

The rejection of claim 1 is incorporated, and further, Heisch discloses:

- ***replacing the target address of the branch instructions with the first address only in functions*** (col. 7, lines 27-31 "Reordered instructions are appended to the end of the original executable (in the "reordered text area") and are replaced (in the "original text area") with branches to the addresses of where the instructions have been moved) ***that are reached during program execution*** (col. 2, lines 65-67 to col. 3, lines 1-3 "TDPR effectively "closes the loop" in the optimization process. It attempts to further optimize a program by collecting information on the actual behavior of a program while it is executed and uses that information to reorder and modify instructions across the entire executable program image to optimize the use of the hardware").

Claim 15 is the apparatus claim corresponding to method claim 1 and rejected under the same rational set forth in connection with the rejection of claim 1 above.

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

7. Claims 3-5 are rejected under 35 U.S.C. 103(a) as being unpatentable over Heisch, in view of admitted prior art.

Per claim 3:

The rejection of claim 1 is incorporated, and further, Heisch discloses:

- ***searching a symbol table for an entry having a symbolic name and reading a linkage stub address associated with the symbolic name*** (see FIG. 10, element 81 and related discussion); and
- ***replacing target addresses of branch instructions having target addresses equal to the linkage stub address*** (col. 7, lines 27-31 "Reordered instructions are appended to the end of the original executable (in the "reordered text area") and are replaced (in the "original text area") with branches to the addresses of where the instructions have been moved) with ***an address at which the code that implements the function is stored*** (col. 5, lines 3-5 "Trace tool 52 determines trace information, such as what functions the instructions perform and the address(es) at which the instructions are stored").

Heisch does not explicitly disclose derivative or having underscore in the name.

However, admitted prior art discloses in an analogous computer system with derivative or having underscore in the name (Applicant's specification, page 2, lines 15-17 "Certain compilers add underscores to names of external functions referenced in an application program. For example, some FORTRAN compilers add underscores to function calls in the application program in order to avoid conflict with FORTRAN "COMMON" blocks").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of derivative or having underscore in the name as taught in prior art into the method of optimizing the executable program code as taught by Heisch. The modification would be obvious because of one of ordinary skill in the art would be motivated have underscore or derivative of the name at the time of execution to differentiate the function name as suggested in prior art (Applicant's specification, page 2, lines 17-19).

Per claims 4 and 5:

The rejection of claim 1 is incorporated, and further, Heisch discloses:

- ***searching a symbol table for an entry having a symbolic name that matches the first name with an*** (see FIG. 10, element 81 and related discussion); and
- ***replacing target addresses of branch instructions having target addresses equal to the linkage stub address*** (col. 7, lines 27-31 "Reordered instructions are appended to the end of the original executable (in the "reordered text area") and are replaced (in the "original text area") with branches to the addresses of where the instructions have been moved) with ***an address at which the code that implements the function is stored*** (col. 5, lines 3-5 "Trace

tool 52 determines trace information, such as what functions the instructions perform and the address(es) at which the instructions are stored").

Heisch does not explicitly disclose derivative or having underscore in the name.

However, admitted prior art discloses in an analogous computer system with derivative or having underscore in the name (Applicant's specification, page 2, lines 15-17 "Certain compilers add underscores to names of external functions referenced in an application program. For example, some FORTRAN compilers add underscores to function calls in the application program in order to avoid conflict with FORTRAN "COMMON" blocks").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of derivative or having underscore in the name as taught in prior art into the method of optimizing the executable program code as taught by Heisch. The modification would be obvious because of one of ordinary skill in the art would be motivated have underscore or derivative of the name at the time of execution to differentiate the function name as suggested in prior art (Applicant's specification, page 2, lines 17-19).

8. Claims 6, 7 and 11 are rejected under 35 U.S.C. 103(a) as being unpatentable over Heisch in view of US Publication No. 2002/0026633 to Koizumi et al., hereinafter called Koizumi.

Per claims 6, 7, and 11:

The rejection of claim 1 is incorporated, and further, Heisch does not explicitly disclose replacing function entry points in the executable program with breakpoints, whereby breakpointed functions are generated; and upon encountering a breakpoint of a breakpointed function during program execution, identifying within the breakpointed function branch instructions that target linkage stub functions.

However, Koizumi discloses in an analogous computer system replacing function entry points in the executable program with breakpoints, whereby breakpointed functions are generated (page 17, paragraph 387 "a break-point setting processing" and "Referring to FIG. 41, the statement ID number in the command is extracted" and "a flag is set which indicates that the break point has been set at the corresponding statement ID number in the correspondence table"); and upon encountering a breakpoint of a breakpointed function during program execution (page 17, paragraph 383 "break point setting command as inputted, a break point setting processing is executed (step 5127), while for an execution command, execution processing is performed"), identifying within the breakpointed function branch instructions that target linkage stub functions (page 17, paragraph 389 "it is checked by consulting the correspondence table whether the break point is set at the extracted address. Unless the break point is set, the processing is repeated, starting from the step 5152, while the processing comes to an end when the break point is set"). Storing of instructions would be inherent if the break point address is being extracted from the memory.

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of setting break point as taught by Koizumi into the method of optimizing the executable program code as taught by Heisch. The modification would be obvious because of one of ordinary skill in the art would be motivated to set a break point within the code so that the translation of the code is appropriate for the target machine as suggested by Koizumi (page 2, paragraph 26 "It is therefore... translated into a machine language... appropriate... target machine... execution...").

9. Claims 8-9 and 12-14 are rejected under 35 U.S.C. 103(a) as being unpatentable over Heisch, Koizumi in view of admitted prior art.

Per claims 8-10 and 12-14:

The rejection of claims 6, 11 is incorporated respectively, and further, Heisch discloses:

- ***searching a symbol table for an entry having a symbolic name and reading a linkage stub address associated with the symbolic name*** (see FIG. 10, element 81 and related discussion); and
- ***replacing target addresses of branch instructions having target addresses equal to the linkage stub address*** (col. 7, lines 27-31 "Reordered instructions are appended to the end of the original executable (in the "reordered text area") and are replaced (in the "original text area") with branches to the addresses of where the instructions have been moved) with ***an address at which the code that implements the function is stored*** (col. 5, lines 3-5 "Trace tool 52 determines trace information, such as what functions the instructions perform and the address(es) at which the instructions are stored").

Neither Koizumi nor Heisch explicitly disclose derivative or having underscore in the name.

However, admitted prior art discloses in an analogous computer system with derivative or having underscore in the name (Applicant's specification, page 2, lines 15-17 "Certain compilers add underscores to names of external functions referenced in an application program. For example, some FORTRAN compilers add underscores to function calls in the application program in order to avoid conflict with FORTRAN "COMMON" blocks").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of derivative or having underscore in the name as taught in prior art into the method of optimizing the executable program code as taught by Heisch. The modification would be obvious because of one of ordinary skill in the art would be motivated have underscore or derivative of the name at the time of execution to differentiate the function name as suggested in prior art (Applicant's specification, page 2, lines 17-19).

Response to Arguments

10. Applicant's arguments with respect to claims 1-15 have been considered but they are not persuasive.

In the remarks, the applicant has argued that:

(i) For claims 1 and 15, Coutant does not teach or suggest that Coutant's process in any way identifies the branch instructions in a program as claimed.

(ii) For claim 2, Coutant does not teach or suggest replacing a target address of a branch instruction only in a function that is reached during program execution.

Examiner's response:

(i) As noted by the Applicants' that Coutant's system is directed to a method for optimizing and efficiently accessing shared data. Coutant's system clearly discloses the branch instructions to transfer control of one point in the code to another (see the rejection above and col. 3, lines 19-34). Further, Applicants' indicated that Coutant's system is not optimizing the code with functions calls and cited various portions of Background and Summary of Coutant. Coutant's system is directed to a method for optimizing and efficiently accessing shared data where shared data could be an executable program or a source program (col. 4, lines 64-67). Also, it is inherent to optimize the function calls in a method to **efficiently** accessing shared data. Applicant only makes general allegations and does not point out any errors in the rejection. Therefore, the rejection is proper and maintained herein.

(ii) For the limitations replacing a target address of a branch instruction only in a function that is reached during program execution as recite in claim 2. Coutant does disclose the replaces the linkage table load with the no-op instructions during the program execution (see col. 8lines 8-35 and FIG. 7 and related discussion) where the addressing is inherent without the address the replacement of instructions is not possible. Applicant only makes general allegations and does not point out any errors in the rejection. Therefore, the rejection is proper and maintained herein.

11. Applicant's arguments with respect to claims 3-14 has been considered but are moot in view of new ground(s) of rejection.

Conclusion

12. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to **Satish S. Rampuria** whose telephone number is **(571) 272-3732**. The examiner can normally be reached on **8:30 am to 5:00 pm** Monday to Friday except every other Friday and federal holidays. Any inquiry of a general nature or relating to the status of this application should be directed to the **TC 2100 Group receptionist: 571-272-2100**

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, **Tuan Q. Dam** can be reached on **(571) 272-3695**. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Satish S. Rampuria
Patent Examiner
Art Unit 2124
06/27/2005

WEI Y. ZHEN
PRIMARY EXAMINER